Hindawi
Journal of Engineering
Volume 2020, Article ID 7286187, 11 pages
https://doi.org/10.1155/2020/7286187

Research Article

Learning Feature Fusion in Deep Learning-Based Object Detector

Ehtesham Hassan,[1] **Yasser Khalil**,[2] **and Imtiaz Ahmad**[3]

[1]*Department of Computer Science and Engineering, Kuwait College of Science and Technology, Kuwait City, Kuwait*
[2]*University of Ottawa, Ottawa, Canada*
[3]*Department of Computer Engineering, Kuwait University, Kuwait City, Kuwait*

Correspondence should be addressed to Ehtesham Hassan; e.hassan@kcst.edu.kw

Received 24 August 2019; Revised 12 April 2020; Accepted 27 April 2020; Published 22 May 2020

Academic Editor: Kevser Dincer

Copyright © 2020 Ehtesham Hassan et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Object detection in real images is a challenging problem in computer vision. Despite several advancements in detection and recognition techniques, robust and accurate localization of interesting objects in images from real-life scenarios remains unsolved because of the difficulties posed by intraclass and interclass variations, occlusion, lightning, and scale changes at different levels. In this work, we present an object detection framework by learning-based fusion of handcrafted features with deep features. Deep features characterize different regions of interest in a testing image with a rich set of statistical features. Our hypothesis is to reinforce these features with handcrafted features by learning the optimal fusion during network training. Our detection framework is based on the recent version of YOLO object detection architecture. Experimental evaluation on PASCAL-VOC and MS-COCO datasets achieved the detection rate increase of 11.4% and 1.9% on the mAP scale in comparison with the YOLO version-3 detector (Redmon and Farhadi 2018). An important step in the proposed learning-based feature fusion strategy is to correctly identify the layer feeding in new features. The present work shows a qualitative approach to identify the best layer for fusion and design steps for feeding in the additional feature sets in convolutional network-based detectors.

1. Introduction

Object detection in natural scenes is an important problem that drives many real-life applications. In the last few decades, a significant amount of research effort has gone into the understanding of object detection problem with general as well as domain-specific challenges [1–5]. As a result, several novel and innovative object detection methods have been developed. Despite continuous efforts from researchers from diverse backgrounds, the present state of the art is far from satisfactory as seen in recent results on standard datasets PASCAL-VOC [6] and MS-COCO [7].

Most recent works on object detection have invariably applied the convolutional neural networks (CNNs) based detector model. These models are learned end-to-end addressing feature extraction, parameter learning, and postprocessing in one- or at most two-stage training process. These methods have significantly improved the state of the art in object detection. The representational capability of features extracted by a CNN depends on the complexity of the detection problem in testing time and variability captured within the training dataset. In real-world applications, a target object's appearance undergoes significant variations due to view angles, lighting, background clutter, and occlusions. Handcrafted features that are designed with domain understanding have often shown to be more distinctive and reliable in many situations. An intelligent fusion of both the modalities of features is expected to achieve better detection performance. In this work, we propose feature enhanced CNN based object detection framework by learning-based fusion of handcrafted features with deep features in the embedding space. We use fundamental color channels: RGB, HSV, and LBP in combination with gradient and orientation histograms to enhance deep features for overcoming the prediction errors due to appearance variations. To effectively combine handcrafted features with deep learning-based features, we also investigate the problem of identifying the optimum layer to inject the handcrafted

features for feature fusion. Our detection framework is based on state-of-the-art YOLO [8] architecture where we inject the handcrafted features at appropriate layer(s) to regularize network weights during the training procedure. We hypothesise that fusing handcrafted features during network learning guides the CNN to extract a more accurate and robust feature set by exploiting upon the complementary information available in the handcrafted feature set. The major contributions in this paper are summarized as follows:

(1) We propose a novel object detection approach based on CNN learning by fusing simple feature descriptors like color channels and gradient histograms [9] by learning-based fusion to train a more robust and accurate detection model.

(2) We use the latest YOLO objection detection architecture as our base. We describe the proposed learning-based feature fusion strategy that uses a qualitative approach to select the best layer for feature injection. In this work, the presented work presents a novel strategy to fuse features in CNN based object detectors.

(3) We demonstrate the comparative performance of the proposed feature fusion approach on PASCAL-VOC and MS-COCO datasets which shows improvement in the original YOLO object detection rate by a significant measure.

The paper is organized as follows: Section 2 presents the relevant previous research works. We briefly discuss the latest YOLO architecture and Integral channel features in Section 4. Section 5 discussed the details and implementation issues of the proposed object detection method. Experimental evaluation of the proposed object detection framework on different datasets is presented in Section 6. We conclude the paper in Section 7 presenting the perspective of our work and future research directions.

2. Related Works

Our work in this paper and the proposed approach therein are related to the CNN based object detection using vision. There has been a significant amount of work in this research area. A thorough review of the literature on object detection and CNN based classification and regression is beyond the scope of this paper. The following discussion reviews the prominent and critical works in the context of the proposed approach for object detection.

Object detection in natural scenes can be pursued using different approaches: model-based, learning-based, and auxiliary [10]. Model-based methods depend on heuristic determined models using color, shape, and intensity attributes. The object characterization in such methods is also referred to as handcrafted features where they are based on the experts' engineering features that best describe representations in images. Learning-based methods are where features from objects are automatically learned from classifiers and are later used for detection. Last, the auxiliary approach is where the location of objects is used as prior information for visualizing them. The auxiliary approach is expensive to deploy as the infrastructure of objects needs to be replaced for them to be effective. Prior knowledge of the object's location reduces the computational search cost and improves accuracy. Furthermore, it helps in eliminating a large number of false positives.

The last few decades of advancements in the state of the art in object detection solutions have seen novel design of handcrafted features such as histogram of gradients (HOG) [11], scale-invariant feature transform (SIFT) [12], and pyramid HOG [13]. These methods used simple discriminative classifiers by scanning the image space, or by feature matching. Jones et al. [14] used integral images for feature extraction for face detection which can be quickly computed. These works generate fixed or variable size object descriptors as a set of local feature vectors. In [15], the authors proposed local-contour based features with two-staged, partially supervised learning for object detection. Among the earliest works on learning-based object detection, the authors in reference [16] proposed learning a sparse part-based object representation. In [17], the authors introduced multiple-component learning-based parts based object detection by automatic learning and combination of individual classifiers. The authors in reference [9] subsequently combined HOG with heterogeneous color channels for pedestrian detection. These features widely known as integral channel features (ICF) again utilize integral images for fast feature computation which is an important requirement for image scan based object detection.

In [18], the authors proposed to learn structure between different object parts using SVM formulation for modeling of object structures at multiple scales using mixtures of deformable part models. An extensive review of pedestrian detection using handcrafted features has been presented by [1]. In [19], the visual attention based modeling is used for salient object detection by using a bootstrap learning model. In [20], the authors proposed regionlets—the integration of different types of features that were computed locally. These features were used to model an object class by a cascaded boosting classifier.

With the recent rise in CNN based learning methods, several CNN based object detection models have been proposed. Girshick et al. [21] proposed region-based convolutional neural networks (R–CNN) as object detection models. The R–CNN model trained independent components for generating region proposals as bounding boxes by using selective search-based image scanning and for classifying the region proposals to one of the object categories. The model was further improvised as Fast R–CNN [22]; nevertheless these models were slow and were difficult to optimize. Faster R–CNN [3] alleviated the difficulties with its previous versions by replacing region proposal network (RPN) with selective search which is trained as a single neural network with fast R–CNN sharing the convolutional feature set. The RPN component of the Faster R–CNN guides the unified network to look into different regions of interest. Faster R–CNN hitherto is considered as the most robust and accurate object detector; these models still lack real-time performance because the first proposals are generated and subsequently proposal labeled in the known categories.

Dai et al. [23] extended the shared, fully convolutional network architecture originally proposed for image segmentation ([24]) to two-stage detection strategy having region proposal and region classification. The end-to-end learning of the network constructs a set of position-sensitive score maps to deal with the translation variance of target objects. The scores are generated by a bank of convolutional layers which encodes the relative spatial position information incorporating translation variance in the learning.

Some recent works on object detection also followed a regression-based approach which includes Single Shot Multibox Detector (SSD) [25], Deconvolutional Single Shot Detector (DSSD) [26], and You Only Look Once (YOLO) [8]. SSD combines the Multibox approach for bounding box regression by using a set of default boxes for predicting the shape offset and category confidence at each location in the image. The authors used VGG-16 as the base network architecture where the network combines predictions from different feature maps with different resolutions. DSSD further improvised SSD by shifting to the "encoder-decoder" type of networks to incorporate context information in the learning by replacing the VGG-16 architecture with residual-101.

YOLO [27], unlike its predecessors, trained single regressor network for direct prediction of object bounding boxes with category confidences. This approach of complete regression without any classification step yielded superior real-time performance in terms of the detection speed. Also, YOLO bettered detection accuracy in comparison with all other detection systems except Faster-RCNN. With the incorporation of anchor boxes for guiding the bounding box prediction in a newly designed network, YOLOv2 bettered all other existing state of the art in PASCAL VOC-2017 challenge. The latest YOLO (here onwards we refer to this version as *YOLOv3*) claims to improve its performance with the incorporation of much deeper network architecture combining predictions at multiple scales. The YOLO algorithms see the entire image all at once through the forward pass of the network which gives more accurate and comprehensive information. This helps the detector in avoiding false positives than the classification based detection systems which focus only on the region proposals. Despite remarkable progress in object detection in natural images due to deep learning-based strategies, the performance of existing models requires improvement from the accuracy and real-time performance standpoint. The latest results on Pascal VOC-challenge [6] and MS-COCO [7] datasets establish that more effort is required to solve the detection problem.

In addition, several CNN based detection models have been designed for specific target objects [28–30]. These methods have raised the benchmark in target objection detection; nevertheless, the problem is far from being solved [5,31–33]. An important distinction in specific object detection is the availability of auxiliary information sources or user feedback which has helped researchers to solve the problem up to the acceptable limit.

In this work, we propose a feature enriched object detector based on the latest YOLO architecture. Our choice of YOLO architecture is based on its comparative detection accuracy and superior detection rate. It is expected that CNN based detection model should be able to learn the object-specific salient features required for detection; nevertheless, it cannot be guaranteed in single objective-based learning formulation. This direction of research exploring learning-based enrichment of deep features by including additional modalities has been not been explored for object detection. In this context, the recent work on foreground extraction for video analysis [34], a deep neural network based framework, exploits on the multistage fusion of the combination of residual features from different convolutional layers. On the other hand, our approach is to feed in the handcrafted features in the network to guide the feature learning process leading towards more accurate detection. For learning-based feature fusion, we use ICF descriptor which has shown remarkable object detection performance as discussed above.

3. Revisiting the Object Detection Problem

Object detection consists of localization of region capturing the object and assigning their labels by processing localized region. The localization problem can be formulated as (i) regression problem that predicts the region of interests or (ii) binary classification problem focusing on detection of foreground region having object segments. The outcome of any of these methods would predict object boundary as set of four coordinates ($x_{\min}$, $x_{\max}$, $y_{\min}$, and $y_{\max}$) as shown in Figure 1.

Label identification of the object region is a N-class classification problem where N denotes the number of classes, for example, {tree, person, bird, dog, bicycle} or {cat, car, tree}. In both the stages of object detection problem, challenges arise because of the underlying variations in lighting conditions, scaling, occlusion, partial view, and orientation. Figure 2 shows some such challenging situations from the COCO dataset.

4. Preliminaries

Before presenting the object detection methodology using learning-based feature fusion, we briefly discuss the integral channel features and *YOLOv3*'s architecture. We also make some modifications in the *YOLOv3* architecture as proposed in [35] which are also described in this section.

4.1. You Only Look Once Object Detector-YOLOv3. The class of YOLO algorithms [8,27,36] look at the entire image when detecting and recognizing objects and extract deep information about classes and their appearance, unlike other approaches such as sliding window-based methods or R–CNN based algorithms. These algorithms treat the detection of objects as a single regression problem giving faster response with a reduction in the design complexity of the detector. Though the significant speed achievement, the algorithms lag in terms of accuracy especially with small objects.

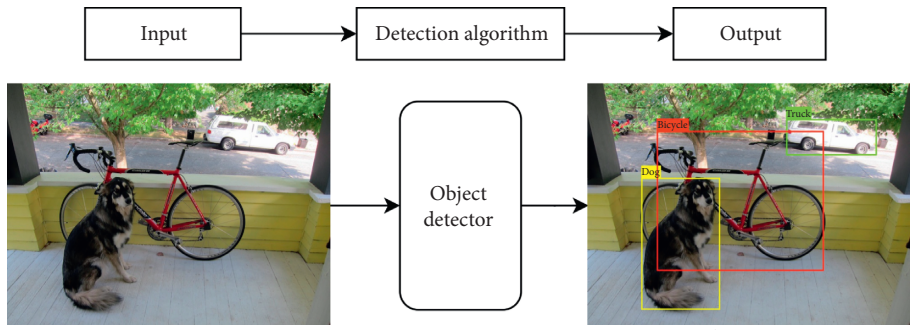

Figure 1: Object detection in an example image.

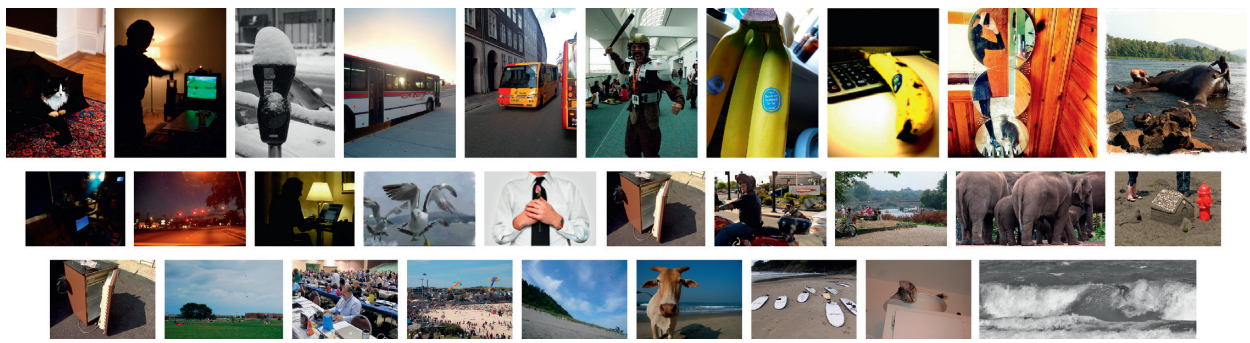

Figure 2: Example images from MS-COCO dataset.

The latest algorithm in the YOLO family, that is, *YOLOv3*, has proven its performance compared to other state-of-the-art detectors. The *YOLOv3* architecture has 107 layers in total distributed as {convolutional = 75; route = 4; residual = 23; upsample = 2; detection = 3}. The architecture uses a new model for feature extraction referred to as Darknet-53. The new model is significantly larger than models used in the earlier version; nevertheless, it has proven to be more efficient than other state of the arts. The model uses 53 convolution layers which takes an input image of size 416×416. Figure 3 shows the architecture of the *YOLOv3* object detector. The Darknet-53 model is pretrained on ImageNet [37]. For the detection task, the network is modified by removing its last layer and then stacking up additional layers resulting in the final network architecture. The first 75 layers in the network represent 52 convolutional layers of the Darknet-53 model pretrained on ImageNet. The remaining 32 layers are added to qualify *YOLOv3* for object detection on different datasets with further training. Besides, *YOLOv3* applies new residual layers similar to skip connections that combine feature maps from two layers using elementwise addition resulting in finer-grained information.

The *YOLOv3* replaces the softmax based activation used in older versions with independent logistic classifiers. Features are extracted using a similar concept to feature pyramid networks. Also, binary cross-entropy loss is now used for class predictions, which is beneficial when faced with images having overlapping labels. The K-means is used for anchor box generation; however, 9 bounding boxes are now used rather than 5. The number of bounding boxes is distributed equally across the three detection scales. In the present design, a route layer is also used which outputs a layer's feature maps.

4.1.1. Prediction. *YOLOv3* processes images by dividing them in $N \times N$ grid, and if the center of the object falls in a grid cell, then that cell is responsible for detecting the object. The network predicts bounding boxes at three different scales. The first detection scale is used for detecting large-sized objects. The second detection scale is used for medium-sized objects and the last for small objects. The three detection layers are displayed in red in Figure 3. Each cell predicts B bounding boxes, and each prediction has 5 predictions: x, y, w, h, and confidence score representing the measure of prediction having an object. The x and y parameters are the center of the box for the grid cell, while w and h are the width and height of the predicted box for the entire image. The confidence score is the Intersection over Union (IoU) between the predicted box and the ground truth. The output prediction is $N \times N \times B \times (5 + C)$ tensor where 5 is the number of predictions per bounding box (x, y, w, h, and the confidence value) and C is the total number of object categories. Figure 4 shows the prediction algorithm of *YOLOv3* for $N \times N$ cells, where each cell has B bounding boxes.

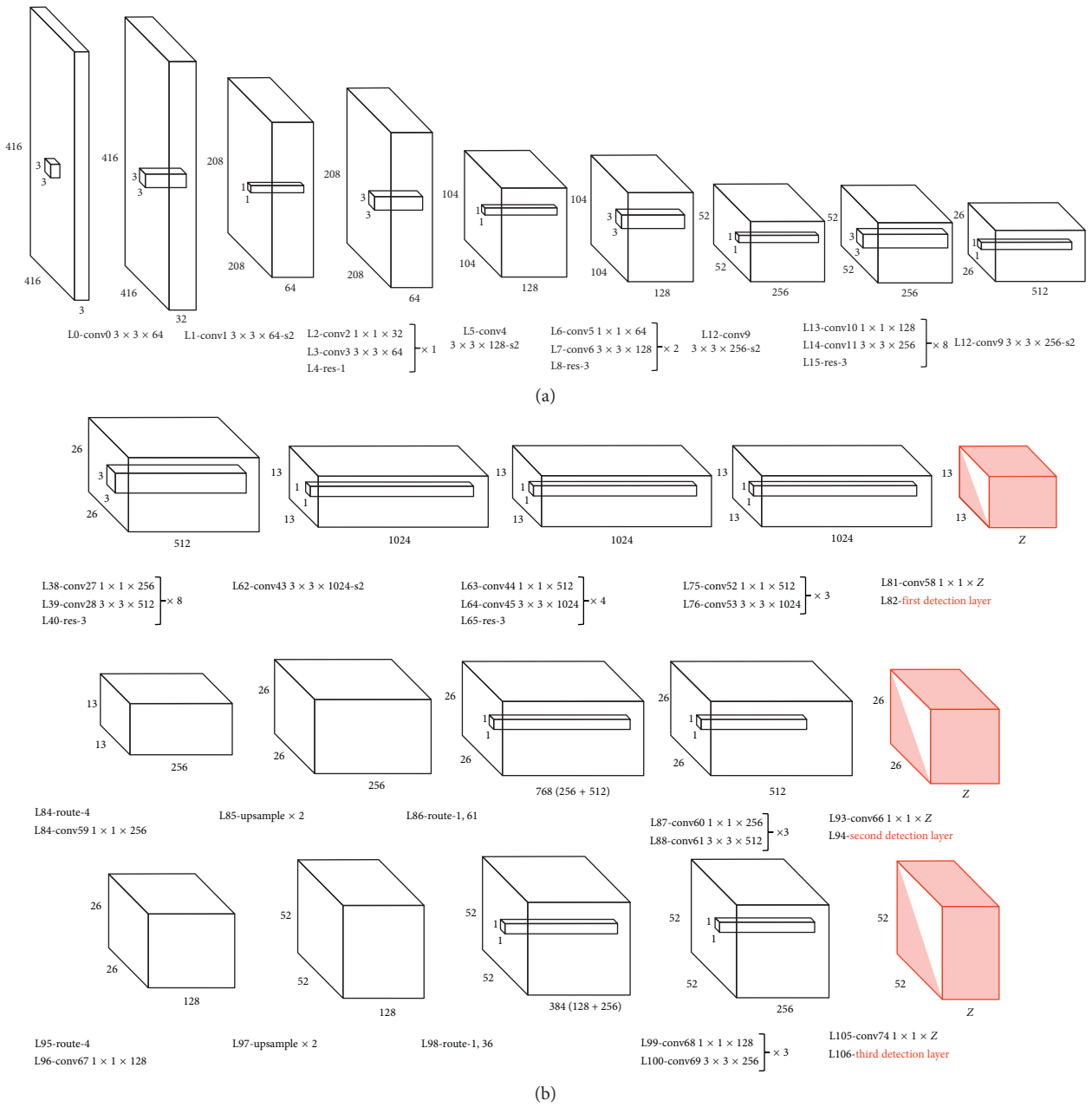

Figure 3: *YOLOv3* network architecture. (a). Convolutional layers in the *YOLOv3*. (b) Detection layers in the *YOLOv3*.

4.1.2. Loss Function. The *YOLOv3* includes a loss function (1) that instructs the network to correctly predict bounding boxes and accurately classify the detected objects with a provision to penalize false positives:

$$\begin{aligned}
&\lambda_{\text{coord}} \sum_{i=0}^{N^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] + \lambda_{\text{coord}} \sum_{i=0}^{N^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w}_i - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h}_i - \sqrt{\hat{h}_i} \right)^2 \right] \\
&+ \sum_{i=0}^{N^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 + \lambda_{\text{noobj}} \sum_{i=0}^{N^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 - \sum_{i=0}^{N^2} \sum_{j=0}^{B} \delta_i \log \left(\hat{p}_i(c) \right) + (1 - \delta_i) \log \left(1 - \hat{p}_i(c) \right).
\end{aligned} \tag{1}$$

The symbols are explained in Table 1. The symbols under hat represent corresponding prediction values. The loss function in the equation has three error components: localization, confidence, and classification as

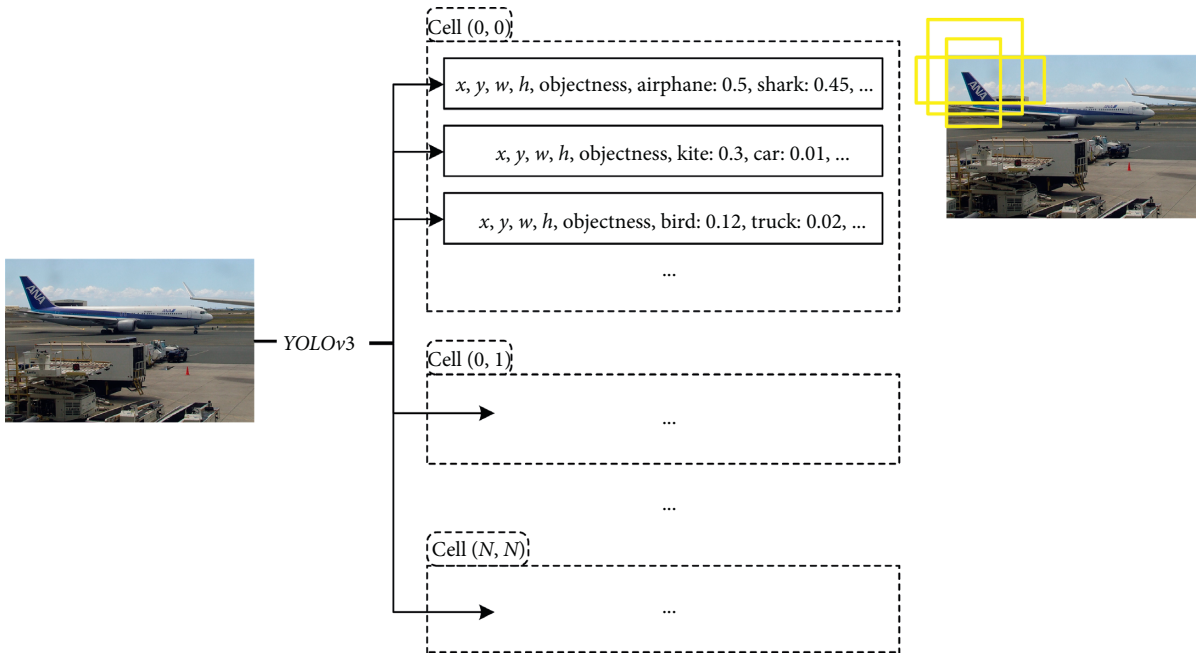

Figure 4: *YOLOv3* prediction output for an example image: cell(i, j) corresponds to the image region within the i^{th} row and j^{th} column of the $N \times N$ grid.

observed in equation (1). Different loss components are combined by sum-squared algorithm as it is easier for optimization. The localization loss is responsible for minimizing the error between the "responsible" bounding box and the ground truth object, if an object is detected in a grid cell.

4.2. Integral Channel Features (ICF). In this work, we fuse different channels of information proposed in the ICF [9] in the *YOLOv3* architecture. ICF descriptor is a series of image channels computed from the input image using linear and nonlinear transformations. A channel refers to a representation of the input image. Next, first-order features are extracted by computing the sum of rectangular regions. The evaluated channels were color channels (RGB, Gray, HSV, and LUV), gradient magnitude, and gradients histogram. The authors in [9] claimed that the most informative channel among all in independent evaluation for pedestrian detection was HOG. Further, a combination of LUV, gradient, and HOG gave the best detection rate. Our proposed work reevaluates the combination of channels because of varying challenges in the present application scenario. In our work, the window size parameter for HOG computation depends on the 2D dimensions of the deep features where they are to be fused with (discussed in Section 4.1). As an example, if the 2D dimensions of a layer where features are to be fused are 13×13 and the input image is of size 416×416, then the window size will be 32×32 so that it results in 13×13 HOGs. For each window, we compute the HOG using six bins. Other HOG parameters including cell and block size used for normalization purposes are also set the same as the window size. The block stride parameter is also set equal to the window size having no overlap between two windows. As shown in Figure 5, the ICF for a given image is the linear concatenation of individual color channels after normalization, the corresponding HOG feature. The preprocessing in feature computation includes the resizing of the input image to align it with the network layer that receives this input. For experiments discussed in this paper, we follow the same steps and parameters for HOG computation as discussed in [9].

5. Learning-Based Feature Fusion in *YOLOv3*

Before we establish the proposed feature fusion, we set out the required steps which will be described subsequently:

(1) Identify the candidate convolution layers for feature injection

(2) Evaluate the layers for feature injection using complete set of ICF channels

(3) Evaluate different combinations of ICF channels using the best network layer obtained in the previous step

(4) Train and test the detector injecting the best ICF channel combination at the best layer position

Our proposal for feature fusion starts with identifying a location and space for handcrafted features within the *YOLOv3* architecture. For injecting an additional set of features in this architecture, we first need to identify the convolution layer to feed additional information. We adopt a qualitative approach to decide on the layer for feature fusion by validating feature fusion on different layers. The depth space of a convolutional layer represents the number of feature maps that hold the deep feature—hereafter

Table 1: Symbols used in *YOLOv3* loss function.

Symbol	Definition
N	Number of grid cells in an image
B	Number of anchor boxes
C_i	Confidence score of j^{th} bounding box in grid cell i
$p_i(c)$	Conditional probability of class c in grid cell i
x_i, y_i, w_i, h_i	Location and size of a bounding box
1_i^{obj}	1, if an object appears in grid cell i; 0, otherwise
1_{ij}^{obj}	1, if an object is present in i^{th} grid cell and the j^{th} "responsible" bounding box; 0, otherwise
$1_{ij}^{\text{noob}j}$	1, if no object is present in i^{th} grid cell and the j^{th} "responsible" bounding box; 0, otherwise
δ_i	1, if predicted label matches the ground truth label; 0, otherwise
λ_{coord}	Constant (default: 5.0)
λ_{noobj}	Constant (default: 0.5)

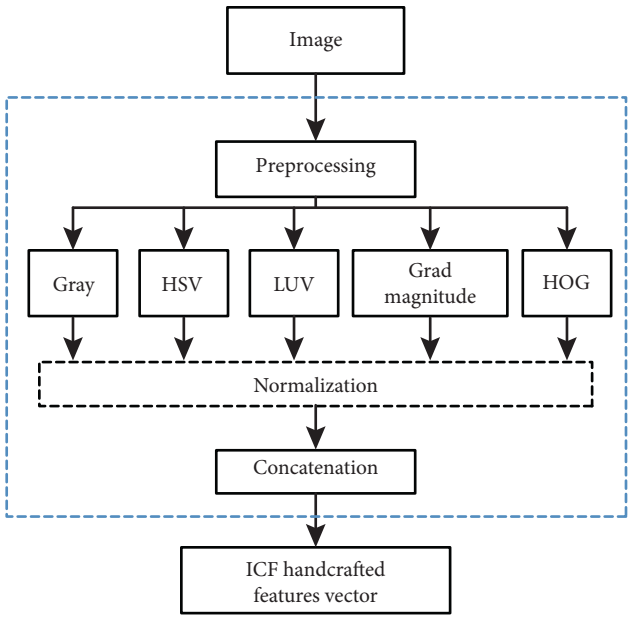

Figure 5: Flowchart for the ICF computation for an example image.

referred to as filters. We need to identify the specific number of filters within the specific layer which will store the additional features. First, we propose to double the original number of filters in the specific layer as shown in Figure 6.

Depending on the number of channels in the ICF set, the same number of filters needs to be reserved for storing handcrafted feature values. The ICF descriptor is injected into the network at the beginning of the training procedure. In the layer selected for ICF injection, the additional filters (introduced by doubling the layer depth) are copied with ICF descriptor from the rear end as shown in Figure 6. The remaining filters are set to zero in the beginning. As the training progresses, the layer weights, that is, filter values update following the learning algorithm. The output generated by this layer depends on the values of all filters. The proposed method for feature fusion diverges from the simple approach of the stacking of handcrafted features with original filters in the selected layer. The extra filters in addition to the filters used for storing ICF descriptor values help regularize the layer weights. We maintain the strategy of doubling the filters for feature fusion regardless of the specific convolutional layer under validation. An example showing the procedure of doubling the number of filters at a specific layer to issue space for ICF fusion is illustrated in Figure 6. In this example, fusion was selected to occur at 62^{nd} layer of *YOLOv3*'s network. The top side of the figure shows a section from the original *YOLOv3*'s network and the bottom side is where the number of filters for the 62^{nd} layer is doubled. The number of filters was changed from 1024 to 2048. The light orange is the new filter depth (2048) and consists of the original filter depth (1024) concatenated with the additional filters of the same depth, represented in green. Additionally, doubling the number of filters in the fusion layer poses fewer design challenges than with filters with direct stacking of handcrafted features.

5.1. Design Issues with Injection of ICF Descriptor. To fuse handcrafted features with deep features, their 2D dimensions have to match the width and height dimensions of the layer. Considering the example in Figure 6 at 62^{nd} layer, if the input image size is 416×416 then the filter dimensions will be 13×13. Hence, all selected channels for fusion should be of size 13×13 to fit into the additional filters. In the testing phase of *YOLOv3*, the size of the input image is fixed at 416×416, so the dimensions of the ICF descriptor at the 62^{nd} layer will always be 13×13. On the other hand, in the training phase of *YOLOv3*, the size of input image changes based on a random parameter in every 10 iterations where the size of the image is defined as a factor of 32, starting from 320×320 up to 608×608. If the input size is 320×320 or 608×608, then at the 62^{nd} layer the filter size becomes 10×10 or 19×19, respectively. The filter size compared with the input image size is downsized by a factor of 32. After making sure that the handcrafted features are of the right size, their values can replace the additional filters in the chosen layer.

The flowchart of the proposed object detector is shown in Figure 7. For training purposes, the desired ICF channels of input images at different scales are calculated offline. These images at different scales are used to incorporate variability in the training process. In the testing phase, ICF channels are calculated only once for each image at a fixed scale. The right side of the figure shows that as our proposed detector is being trained/tested fusion takes place at the layer named ``n^{th} *YOLOv3* convolutional layer." The doubled number of

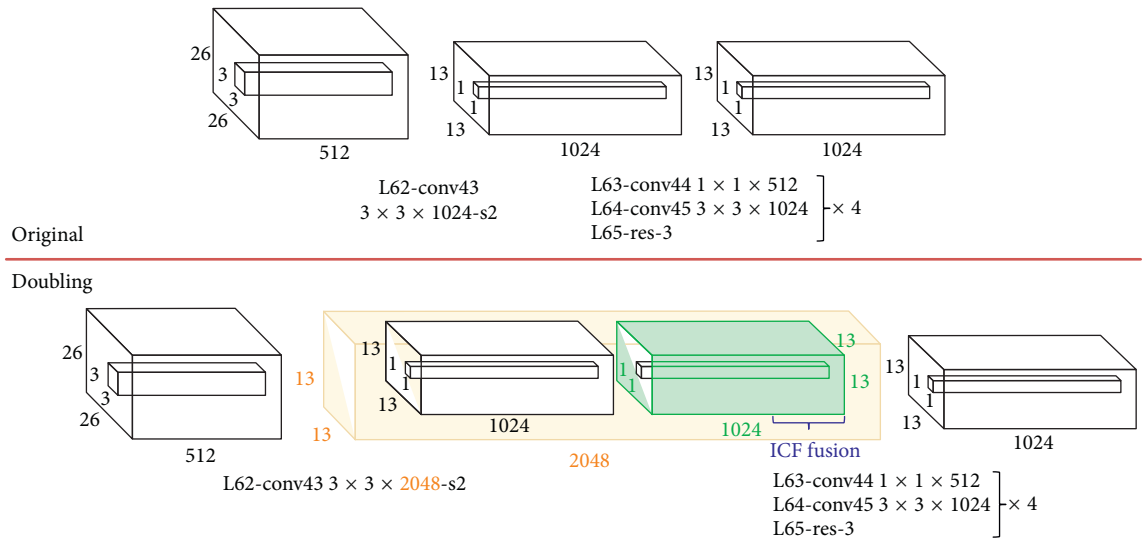

Figure 6: ICF descriptor injection in the specific convolutional layer.

filters and the number of ICF channels are represented as $2x$ and y, respectively. The figure shows how the ICF overlaps with a portion of the deep features residing in some of the additional filters.

6. Experimental Evaluation

A preliminary evaluation of the proposed framework is performed on the PASCAL-VOC dataset (VOC2007 detection task). The original *YOLOv3* detector achieved the mAP of 67.5% on the testing set with input image dimension as 416×416. The *YOLOv3* implementation available by [35] is used for benchmarking our experimental results. As mentioned in Section 5, we first identify the convolutional layer for injecting the handcrafted features, and the required space (i.e., the number of filters). First, we figure out the best layer location to fuse ICF channels. Subsequently, by another set of experiments, we determine the best combination of ICF channels. We separate a validation set comprising one-tenth of the training examples by random selection. The validation set is used for testing the convolutional layers and combination of ICF channels for optimum selection. For determining the best layer for fusion, we use all channels of information in ICF as used in the original work by Dollar et al. [9]. This is done by doubling the number of filters at the layer under evaluation, fusing 17 channels of ICF set, and checking the resulting mAP. We evaluated the two convolutional layers preceding a detection layer in *YOLOv3* architecture. For each experiment, the network is trained for 25,000 iterations. The remaining training parameters are set as the original *YOLOv3* detector. We performed the first evaluation on 80^{th} layer which is the last convolutional layer before the detection layer.

Doubling the number of filters for 80^{th} layer and injecting the full set of ICF channels resulted in the mAP of 69.8%. In comparison with the benchmark performance based on *YOLOv3* performance, we achieved an increase of 1.7% on mAP scale. The next experiment on 79^{th} layer achieved the mAP of 71.5%, which is more effective than the previous experiment. Table 2 lists the mAP scores for experiments conducted for determining the best layer position for feature fusion. As observed, the best mAP score of 71.5% is achieved at 79^{th} layer.

After determining the most suitable layer for ICF fusion, we conduct experiments to discover the combination of ICF channels for fusion that is most suitable for the object detection task in the VOC dataset. Again, in this set of experiments, we fuse ICF channels by doubling the number of filters at the 79^{th} layer. In this experiment, we run the training process for 50,000 iterations. Fusing all ICF channels, color, gradient magnitude, and HOG, achieved the mAP of 77.7%. The entire ICF descriptor summed up to 17 channels, considering 6 channels for HOG. Removing the RGB channel from the 17 ICF channels increased the mAP to 78.2% showing a minuscule increase of 0.5% from the preceding experiment. Considering original input images are in RGB color values, the increase is not significant. We also explored other channel combinations; the corresponding mAP scores are presented in Table 3.

As seen, the best ICF channel combination for VOC2007 detection task consists of Gray, HSV, LUV, gradient magnitudes and HOG achieving maximum mAP score of 79.1%. We retrain the *YOLOv3* detector on complete training set fusing the finalized channels of ICF descriptor. The network achieved the mAP of 78.7% on the testing set which is 11.2% more than the detection rate achieved by original *YOLOv3* detector.

6.1. Evaluation on MS-COCO Dataset. MS-COCO dataset consists of 82,783 training, 40,504 validation, and 40,775 testing images belonging to 80 categories. In the literature, the *YOLOv3* [8] reported the mAP of 55.3% on the COCO dataset for an input image size of 416×416. All experiments reported in this work were executed on a NVIDIA GeForce GTX 1080 GPU desktop which is unsubstantial in

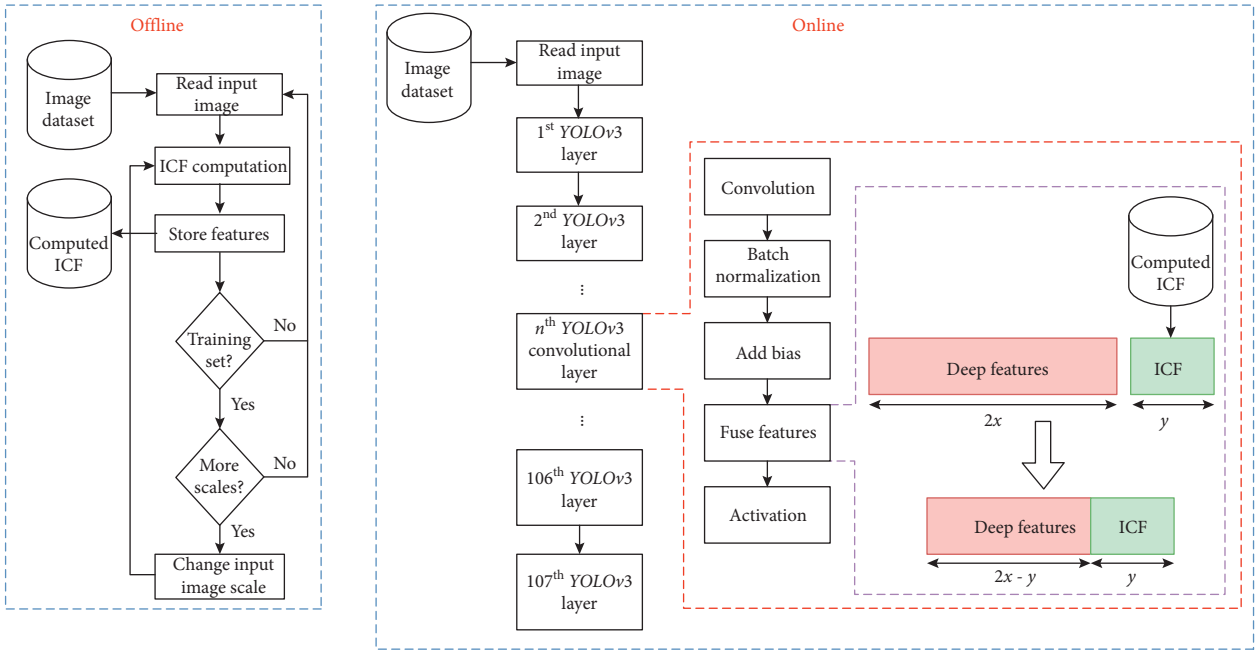

Figure 7: Flowchart of the proposed object detector.

comparison with the infrastructure used in the original paper [8]. Therefore, we adopt an alternate strategy for benchmarking where the YOLO detector (original and with proposed fusion strategy) is retrained on the MS-COCO dataset and we compare the detection performance at 50000 iterations. The original *YOLOv3* evaluation on COCO testing dataset achieved the mAP of 36.5% for 50,000 iterations.

After determining the best location for ICF fusion using the mAP values achieved on the validation set as shown in Table 4, we performed experiments to determine the best combination of ICF channel fusion that best suits the COCO dataset (Table 5). As shown in Table 4, doubling the 78^{th} layer resulted in the highest mAP score (37.3%). Therefore for all subsequent experiments related to the COCO dataset were conducted with the doubled number of filters at the 78^{th} layer. All experiments were evaluated at 50,000 iterations. Fusing all ICF channels (color, gradient magnitude, and HOG) resulted in the mAP of 37.7%. Removing RGB from the 17 ICF channels led to the mAP of 37.3%. Other channel combinations were tried out but did not give good results such as fusing only HOG features (6 channels), LUV + HOG (9 channels), gray + LUV + gradient + HOG (11 channels), and gray + HSV + LUV + gradient (8 channels). The best combination that gave the highest mAP score consisted of fusion of the all ICF channels.

Furthermore, the evaluation on testing set with fusion of the complete set of ICF channels at the 78^{th} layer achieved 37.8% of the mAP value. With COCO dataset, we also experimented with the effect of normalization on the ICF channels. To measure the effect of normalization on the performance of the network, we tried out two different normalization techniques. All the results discussed so far were achieved without any normalization.

Table 2: VOC dataset: detection rate with fusion of complete set of ICF channels.

Layer number	Number of filters after doubling	mAP
59	512	66.1
60	1024	68.4
62	2048	66.2
77	1024	69.8
78	2048	70.9
79	1024	71.5
80	2048	69.8

Table 3: VOC dataset: detection rate with fusion of ICF channel combinations at the 79^{th} layer.

ICF channel combination	Number of channels	mAP
RGB + gray + HSV + LUV + grad. + HOG	17	77.7
Gray + HSV + LUV + grad. + HOG	14	77.4
HSV + LUV + grad. + HOG	13	78.6
Gray + LUV + grad. + HOG	11	78.0
LUV + grad. + HOG	10	79.1
Gray + HSV + grad. + HOG	11	76.8
Gray + HSV + LUV	7	76.2
Gray + HSV + LUV + HOG	13	77.1

We first experimented with max normalization technique dividing each channel by its maximum range so that all values fall in the range of 0 to 1. The fusion of max normalized ICF channels in *YOLOv3* achieved the mAP of 38.4% on the testing set which is 0.6% higher than the best result achieved with unnormalized features and 1.9% with the original *YOLOv3* performance. This confirms with the training principle of deep networks which suggests the use of input normalization as one of the tools

Table 4: COCO dataset: detection rate with fusion of complete set of ICF channels.

Layer number	Number of filters after doubling	mAP
57	1024	35.2
59	512	33.7
60	1024	35.6
73	2048	35.1
75	1024	36.6
77	1024	36.3
78	2048	37.3
79	1024	36.9

Table 5: MS-COCO : detection rate with fusion of ICF channel combinations at the 78^{th} layer.

ICF channel combination	Number of channels	mAP
RGB + gray + HSV + LUV + grad. + HOG	17	37.7
Gray + HSV + LUV + grad. + HOG	14	37.3
HSV + LUV + grad. + HOG	13	37.4
Gray + LUV + grad. + HOG	11	36.4

for performance improvement. In the second normalization technique, we used Z-score based normalization which converts data in all channels to have mean 0 and standard deviation 1. However, the Z-score normalization did not perform as good as the previous normalization technique.

7. Conclusion and Future Work

In this work, we present an approach to fuse handcrafted features in the convolutional neural network based object detector. The fusion of handcrafted features with learning-based features has proved to be effective in many earlier works. In this work, we demonstrate the methodological steps to fuse handcrafted features in the latest version of the YOLO detector. Our experiments with a combination of simple integral channel features fusion in YOLO have yielded substantial improvement in the detection rate on PASCAL-VOC and MS-COCO datasets. In conventional machine learning, early, late, and learning-based fusion have been primary strategies for feature fusion. However, deep learning networks are designed for specific input sizes which pose a challenge for algorithm designer to feed in extra information in the network unless the network is redesigned. The work presented a novel approach based on methodological steps to address both the problems. In future work, we plan to explore the approach for sequence-based learning for object detection and tracking.

Data Availability

This work utilizes standard datasets which are open access.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: an evaluation of the state of the art," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 743–761, 2012.

[2] D. T. Nguyen, W. Li, and P. O. Ogunbona, "Human detection from images and videos: a survey," *Pattern Recognition*, vol. 51, pp. 148–175, 2016.

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[4] S. Zafeiriou, C. Zhang, and Z. Zhang, "A survey on face detection in the wild: past, present and future," *Computer Vision and Image Understanding*, vol. 138, pp. 1–24, 2015.

[5] S. Zhang, R. Benenson, M. Omran, J. Hosang, and B. Schiele, "How far are we from solving pedestrian detection?" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, NV, USA, pp. 1259–1267, June 2016.

[6] Pascal visual object classes homepage (pascal-voc): http://host.robots.ox.ac.uk/pascal/VOC/.

[7] Coco-common objects in context (ms-coco): http://cocodataset.org/.

[8] J. Redmon and A. Farhadi, *Yolov3: An Incremental Improvement*, 2018, https://arxiv.org/abs/1804.02767.

[9] P. Dollár, Z. Tu, P. Perona, and S. Belongie, *Integral Channel Features*, BMVC Press, Swansea, UK, 2009.

[10] F. Arman and J. K. Aggarwal, "Model-based object recognition in dense-range images-a review," *ACM Computing Surveys (CSUR)*, vol. 25, no. 1, pp. 5–43, 1993.

[11] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *Computer Vision and Pattern Recognition*, vol. 1, pp. 886–893, 2005.

[12] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[13] A. Bosch, A. Zisserman, and X. Munoz, "Representing shape with a spatial pyramid kernel," in *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*, pp. 401–408, Association for Computing Machinery, Amsterdam, The Netherlands, July 2007.

[14] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[15] J. Shotton, A. Blake, and R. Cipolla, "Contour-based learning for object detection, in Computer Vision," in *Proceedings of the Tenth IEEE International Conference on Computer Vision*, vol. 1, IEEE, Beijing, China, pp. 503–510, October 2005.

[16] S. Agarwal and D. Roth, "Learning a sparse representation for object detection," in *Proceedings of the European Conference on Computer Vision*, pp. 113–127, Copenhagen, Denmar, May 2002.

[17] P. Dollár, B. Babenko, S. Belongie, P. Perona, and Z. Tu, "Multiple component learning for object detection," in *Proceedings of the 10th European Conference on Computer*, vol. 10, Springer, Berlin, Germany, pp. 211–224, June 2008.

[18] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[19] N. Tong, H. Lu, X. Ruan, and M.-H. Yang, "Salient object detection via bootstrap learning," in *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition, IEEE, Boston, MA, USA, pp. 1884–1892, June 2015.

[20] X. Wang, M. Yang, S. Zhu, and Y. Lin, "Regionlets for generic object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, IEEE, Sydney, NSW, Australia, pp. 17–24, December 2013.

[21] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Columbus, OH, USA, pp. 580–587, June 2014.

[22] R. Girshick, "Fast R-Cnn," in *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448, Santiago, Chile, 2015.

[23] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: object detection via region-based fully convolutional networks," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 379–387, Barcelona, Spain, December 2016.

[24] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Boston, MA, USA, pp. 3431–3440, June 2015.

[25] W. Liu, D. Anguelov, D. Erhan et al., "SSD: single shot multibox detector," in *European Conference on Computer Vision*, pp. 21–37, Amsterdam, Netherlands, 2016.

[26] C. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, *Deconvolutional Single Shot Detector*, 2017, https://arxiv.org/abs/1701.06659.

[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Las Vegas, NV, USA, pp. 779–788, June 2016.

[28] Y. Tian, P. Luo, X. Wang, and X. Tang, "Deep learning strong parts for pedestrian detection," in *Proceedings of the IEEE International Conference on Computer Vision*, IEEE, Santiago, Chile, pp. 1904–1912, December 2015.

[29] Y. Tian, P. Luo, X. Wang, and X. Tang, "Pedestrian detection aided by deep learning semantic tasks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Boston, MA, USA, pp. 5079–5087, June 2015.

[30] Y. Zhou, L. Liu, L. Shao, and M. Mellor, "Dave: a unified framework for fast vehicle detection and annotation," in *Proceedings of the European Conference on Computer Vision*, pp. 278–293, Amsterdam, The Netherlands, October 2016.

[31] B. Li, T. Zhang, and T. Xia, "Vehicle detection from 3d lidar using fully convolutional network," 2016, http://arxiv.org/abs/1608.07916.

[32] L. Zhang, L. Lin, X. Liang, and K. He, "Is faster r-cnn doing well for pedestrian detection?" in *Proceedings of the European Conference on Computer Vision*, pp. 443–457, Amsterdam, The Netherlands, October 2016.

[33] S. Zhang, R. Benenson, M. Omran, J. Hosang, and B. Schiele, "Towards reaching human performance in pedestrian detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 973–986, 2018.

[34] T. Akilan, Q. M. J. Wu, and W. Zhang, "Video foreground extraction using multi-view receptive field and encoder-decoder dcnn for traffic and surveillance applications," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 10, pp. 9478–9493, 2019.

[35] A. B. Alexey, *Windows and Linux Version of Darknet Yolo V3 & V2 Neural Networks for Object Detection*, pp. 6–2, 2018.

[36] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Seattle, Washington, USA, pp. 6517–6525, July 2017.

[37] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.